File I/O - Reading - Individual Exercises

The purpose of this exercise is to provide you with the opportunity to create command line applications that can read and analyze files.

Learning Objectives

After completing this exercise, students will understand:

- How to provide arguments to a command line application from the console.
- How to read data from a text file.
- How to create complex, interactive command line applications that are data-driven.
- How to handle file paths provided as application input.
- How to read, interpret, and resolve errors related to file I/O.

Evaluation Criteria & Functional Requirements

Your code will be evaluated based on the following criteria:

- The project must not have any build errors.
- The expected results are returned from the application.
- Paths to the input files are not hard coded—that is, you should be able to pass the path to the input file to the application.

WordSearch

Part 1

Write a program that asks the user for a search string and a filesystem path for a text file. When the problem is run, it searches the file for occurrences of the search string and each time it finds a line that matches, it displays the line number and contents of the line it was found in on the console.

NOTE: Line numbers begin with 1.

Use the `alices_adventures_in_wonderland.txt` file for test input. **The path to the file should be specified by the user and should not be hard coded in the class.**

For instance, if you wanted to run the program from the command line, you should be able to specify the file by running the application with the command at the root of the project (the same directory this README file is in):

```
mvn exec:java -Dexec.mainClass="com.techelevator.WordSearch"
What is the file that should be searched?
[path-to-the-file]
What is the search word you are looking for?
dog
604) conversation. "Are you--are you fond--of--of dogs?" The Mouse did not
605) answer, so Alice went on eagerly: "There is such a nice little dog near
```

```
617) won't talk about cats or dogs either, if you don't like them!"
622) history, and you'll understand why it is I hate cats and dogs."
1728) "To begin with," said the Cat, "a dog's not mad. You grant that?"
1732) "Well, then," the Cat went on, "you see a dog growls when it's angry,
```

Part 2

Modify the WordSearch program to ask the user if the search should be case insensitive.

For example:

```
mvn exec:java -Dexec.mainClass="com.techelevator.WordSearch"
What is the file that should be searched?
[path-to-the-file]
What is the search word you are looking for?
Who
Should the search be case sensitive? (Y\N)
Y
204)           Who stole the Tarts?                          140
518) shall only look up and say, 'Who am I then? Tell me that first, and
1017) "An arm, you goose! Who ever saw one that size? Why, it fills the whole
1042) fancy--Who's to go down the chimney?--Nay, _I_ sha'n't! _You_ do
1169) "Who are _you_?" said the Caterpillar.
1200) "You!" said the Caterpillar contemptuously. "Who are _you_?"
1452) open place, with a little house in it about four feet high. "Whoever
2038) "Who's making personal remarks now?" the Hatter asked triumphantly.
2204) at her, and the Queen said severely, "Who is this?" She said it to the
2371) "Who _are_ you talking to?" said the King, coming up to Alice, and
3042)           Who for such dainties would not stoop?
3050)           "Beautiful Soup! Who cares for fish,
3052)           Who would not give all else for two
3080) [Sidenote: _Who Stole the Tarts?_]
3115) [Illustration: _Who stole the tarts?_]
3435) "Who is it directed to?" said one of the jurymen.
3567) "Who cares for _you_?" said Alice (she had grown to her full size by
```

QuizMaker (Challenge)

Create a quiz maker program that asks the user a question. The user should be presented with several multiple choice answers and be allowed to specify the correct answer.

The program should read the questions from an input file during startup. The questions and answers in the file are pipe-delimited ("|") and correct answers are marked with an asterisk ("*") in the file.

For example:

```
Question-1|answer-1|answer-2|correct-answer*|answer-4
```

An example of the file might look something like this:

```
What color is the sky?|Yellow|Blue*|Green|Red
What are Cleveland's odds of winning a championship?|Not likely*|Highly likely|Fat
chance|Who cares??
```

The application is started with the command `mvn exec:java -Dexec.mainClass="com.techelevator.QuizMaker"`.

Tips

- Create a class that can hold a quiz question, its available answers, and the correct answer.
- Try holding each quiz question in a list of quiz questions.

Part 1

Ask a single question to the user when the application is opened. Don't show the asterisk in the list of choices.

Example

```
Where is the quiz file?
[path-to-quiz-file]

What color is the sky?
1. Yellow
2. Blue
3. Green
4. Red

Your answer: 2
Correct!
```

Part 2

Go through all of the available quiz questions and ask the user each one sequentially. Record how many answers they got correct and print out the total at the end.

Example

```
Where is the quiz file?
[path-to-quiz-file]

What color is the sky?
1. Yellow
```

```
2. Blue
3. Green
4. Red

Your answer: 3
Sorry that isn't correct!

What are the Cleveland Browns' odds of winning a championship?
1. Not likely
2. Highly likely
3. Fat chance
4. Who cares??

Your answer: 1
Correct!

You got 1 answer(s) correct out of the total 2 questions asked.
```

Getting Started

- Import the file-io-part1-exercises project into Eclipse.
- Open the java file for the application you're working on. The files are located in the src/main/java/com/techelevator directory.
- Provide enough code to get the program started.
- Verify your work on the command line.
- Repeat until the features have been implemented as outlined in the functional requirements.

Tips and Tricks

- Command line applications can be a valuable asset to the teams that you work on. There will be occasions when routine tasks need to be completed with files at some scheduled intervals. Learning how to create applications that can load data into systems, provide alerts when data is not correct, or even automate other systems is an essential skill for developers to gain.
- What tasks might you be able to automate in your own life by applying the knowledge you now have about reading data from files?
- Why might it be important to allow people to pass their own file path to the applications you write? How else might you get the path without explicitly asking your customer to provide the path?
- Learning how to read errors that occur when your applications fail is an important skill to develop as an application developer. When an error occurs, reading the error details is a great way to start the discovery process for diagnosing the problems that occurred.
- Maven is a great tool for Java developers to learn. This tool can be used to run your applications, build and compile code, run your tests, and even create new projects. The best part of command line utilities is that they can often be automated without the need to install full versions of the software. Take some time to learn about the powerful features Maven provides.
- An important skill to learn as a developer is how to read documentation related to the language you're working with. The documentation for the Java File class may be valuable as you work on this exercise.
